Final

D. Hardt
J. Bufu
Sxip Identity
J. Hoyt
JanRain

December 5, 2007

TOC

OpenID Attribute Exchange 1.0 - Final

Abstract

OpenID Attribute Exchange is an OpenID service extension for exchanging identity information between endpoints. Messages for retrieval and storage of identity information are provided.

Table of Contents

TOC

1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in **[RFC2119]**.

1.1. Definitions and Conventions

User:
: Also referred to as "End User" or "Subject". A person with a digital identity who participates in OpenID-based identity information exchanges using their client software, typically a web browser.

Identity Data:
: A property of a digital identity in which the Property Name and Property Value are represented as a name-value pair.

Attribute
: The base of the information model used to describe the Identity Data, for the purpose of exchanging it.

Persona:
: A subset of the user's identity data. A user can have multiple personas as part of their identity. For example, a user might have a work persona and a home persona.

OpenID Provider:
: Also called "OP" or "Server". An OpenID Authentication server on which a Relying Party relies for an assertion that the end user controls an Identifier.

Relying Party:
: Also called "RP" or "Consumer". A Web application that wants proof that the end user controls an Identifier, and requests identity data associated with the end user.

All OpenID Attribute Exchange messages MUST contain the following extension namespace declaration, as specified in the Extensions section of OpenID-Authentication-2.0:

```
openid.ns.<extension_alias>=http://openid.net/srv/ax/1.0
```

The actual extension namespace alias should be determined on a per-message basis by the party composing the messages, in such a manner as to avoid conflicts between multiple extensions. For the purposes of this document, the extension namespace alias for the attribute exchange service will be "ax".

2. Overview

The attribute exchange service extension is identified by the URI "http://openid.net/srv/ax/1.0". This URI MUST be specified in the extension namespace declaration.

An attribute is a unit of personal identity information that is identified by a unique URI. It may refer to any kind of information. A reference example of defining attribute types is provided by **[OpenID.axschema]**.

This service extension defines two message types for transferring attributes: fetch (see **Section 5**) and store (see **Section 6**). Fetch retrieves attribute information from an OpenID Provider, while store saves or updates attribute information on the OpenID Provider. Both messages originate from the Relying Party and are passed to the OpenID

Provider via the user agent as per the OpenID Authentication protocol specification.

The request parameters detailed here MUST be sent using the **[OpenID.authentication-2.0]** extension mechanism.

TOC

3. Information Model

The OpenID Attribute Exchange service extension provides a mechanism for moving identity information between sites, as such its information model is simple:

An attribute is associated with a Subject Identifier

An attribute has a type identifier and a value

An attribute type identifier is a URI

An attribute value can be any kind of data.

TOC

3.1. Subject Identifier

An identifier for a set of attributes. It MUST be a URI. The subject identifier corresponds to the end-user identifier in the authentication portion of the messages. In other words, the subject of the identity attributes in the attribute exchange part of the message is the same as the end-user in the authentication part. The subject identifier is not included in the attribute exchange.

TOC

3.2. Attribute Type Identifier

An attribute type identifier MUST be a URI, which is used for referring to property values.

If an attribute type identifier URI can be resolved then it MAY be dereferenced to retrieve a description of the property. OpenID Providers can use the metadata obtained through dereferencing new or unknown attribute types to dynamically assist the user in providing the attribute.

This provides for flexibility and extensibility. Flexibility in that both URNs and URLs can be used to refer to property values. Extensibility allows any individual site, or consortium of sites, to define their own attribute types with agreements on the syntax and semantics of their associated attribute values.

[OpenID.axschema] outlines an example method of defining new attribute type URIs, and also provides a set of attribute types with their associated metadata schema and data formats.

TOC

3.3. Attribute Value

A attribute value MUST be a **UTF-8** [RFC3629] string. In order to comply with the data formats defined by the underlying **[OpenID.authentication-2.0]** protocol, attribute values MUST NOT contain newlines (UCS codepoint 10, "\n").

OpenID Attribute Exchange can be used to transfer any kind of data. If the data contains newlines, is not a UTF-8 string or it is so desired by the parties transferring the data, the data MUST be encoded to a UTF-8 string without newlines.

TOC

3.3.1. Attribute-Specific Encodings

Attribute-specific encodings can be defined using the attribute metadata descriptions and may be applied by the protocol layer above OpenID Attribute Exchange.

Optionally, attribute-specific encodings may use language tags **[OpenID.value-lang-1.0]** for localization.

TOC

4. Discovery

Discovery of the attribute exchange service extension is achieved via the mechanism described in **[OpenID.authentication-2.0]**. The attribute exchange namespace "http://openid.net/srv/ax/1.0" SHOULD be listed as an <xrd:Type> child element of the <xrd:Service> element in the XRDS discovery document.

TOC

5. Fetch Message

The fetch message is used to retrieve personal identity attributes from an OpenID Provider.

TOC

5.1. Fetch Request Format

With the exception of "openid.ax.mode", all of the following request fields are OPTIONAL, though at least one of "openid.ax.required" or "openid.ax.if_available" MUST be specified in the request, and any attribute alias present in a "openid.ax.required" or "openid.ax.if_available" parameter MUST have an associated "openid.ax.type.<alias>" parameter. The supported length for attribute aliases MUST be at least 32 characters.

Multiple attribute aliases in the "openid.ax.required" and "openid.ax.if_available" directives are separated with a comma, ",".

openid.ax.mode

REQUIRED. Value: "fetch_request".

openid.ax.type.<alias>

The value of this parameter specifies the type identifier URI of a requested attribute. The <alias> will further be used to identify the attribute being exchanged.

Attribute aliases MUST NOT contain newline and colon characters, as specified in the Data Formats / Protocol Messages section of **[OpenID.authentication-2.0]**; they also MUST NOT contain commas (",") and periods (".").

openid.ax.required

Value: an attribute alias, or a list of aliases corresponding to the URIs defined by "openid.ax.type.<alias>" parameters. Multiple attribute aliases are separated with a comma, ",".

By requesting attributes using this field, a hint is sent to the OP about the RP's requirements for offering certain functionality and should be used by the OP to help the user decide what attributes to release. RP's requirements should not be enforced by the OP.

The RP should offer, out of band of attribute exchange, an alternate method of collecting the attributes it needs, if they weren't obtained via attribute exchange.

openid.ax.if_available

Value: an attribute alias, or a list of aliases corresponding to the URIs defined by "openid.ax.type.<alias>" parameters. Multiple attribute aliases are separated with a comma, ",".

Attributes requested using this field are deemed optional by the RP; the RP should be able to complete the interaction with the user even if values are not provided by the OP for the optional attributes.

openid.ax.count.<alias>

The number of values for the specified attribute alias the Relying Party wishes to receive from the OpenID Provider. If present, the value MUST be greater than zero, or the special value "unlimited" which signifies that the RP is requesting as many values as the OP has for the attribute. If absent, exactly one value is requested.

OpenID Providers MAY return less than or the exact number of values speficied by this field for the associated attribute, but MUST NOT return more than the number of requested values for the attribute.

openid.ax.update_url

If present, the OpenID Provider may re-post the fetch response message to the specified URL at some time after the initial response has been sent, using a OpenID Authentication Positive Assertion. If the OpenID Provider supports this feature it MUST return the parameter as part of the fetch response message. If it does not support this feature it may legally ignore this parameter.

The value of the "openid.ax.update_url" field MUST be used as value for "openid.return_to" field of the underlying OpenID Authentication Positive Assertion of the fetch response update.

The "openid.ax.update_url" value MUST also match the realm specified in the underlying OpenID message of the fetch request, if a "openid.realm" field is present. The matching rules are the ones specified in the "Realms" section of the OpenID Authentication protocol.

This "unsolicited" response message would be generated in response to an attribute information update, and would contain the updated data. The OP should obtain the user's consent for resending the updated data to the RPs, as with any OpenID Positive Assertion.

The relying party may include transaction data encoded in the URL such that it contains enough information to match the attribute information to the identity subject. Additional information may be encoded in the URL by the relying party as necessary.

If an RP wishes to receive no further updates for an attribute, it MAY return the HTTP 404 response code to the corresponding "update_url". OPs MAY decide to stop sending updates after encountering 404 response codes.

This example requests the required full name and gender information, and the optional favourite dog and movie information. The Relying Party is interested in up to three favorite movies associated with the subject identifier.

```
openid.ns.ax=http://openid.net/srv/ax/1.0
openid.ax.mode=fetch_request
openid.ax.type.fname=http://example.com/schema/fullname
openid.ax.type.gender=http://example.com/schema/gender
openid.ax.type.fav_dog=http://example.com/schema/favourite_dog
openid.ax.type.fav_movie=http://example.com/schema/favourite_movie
openid.ax.count.fav_movie=3
openid.ax.required=fname,gender
openid.ax.if_available=fav_dog,fav_movie
openid.ax.update_url=http://idconsumer.com/update?transaction_id=a6b5c41
```

TOC

5.2. Fetch Response Format

The fetch response message supplies the information requested in the fetch request. Each attribute is supplied with the assigned alias prefixed by "openid.ax.value." as the lvalue and the attribute value as the rvalue. Attribute types are also returned in the "openid.ax.type.<alias>" parameters. The supported length for attribute aliases MUST be at least 32 characters.

With the exception of "openid.ax.mode", all of the following request fields are OPTIONAL, though any attribute value present in a "openid.ax.value.<alias>" parameter MUST have an associated "openid.ax.type.<alias>" parameter.

If a value was not supplied or available from the user, the associated "openid.ax.value.<alias>" field SHOULD NOT be included by the OP in the fetch response. An "openid.ax.count.<alias>" with a value of "0" together with its corresponding "openid.ax.type.<alias>" field MAY be included to explicitly state that no values are provided for an attribute.

Validation of the received data should be performed out of band of attribute exchange by the RP.

openid.ax.mode

REQUIRED. Value: "fetch_response".

openid.ax.type.<alias>

The value of this parameter specifies the type identifier URI for an attribute in the fetch response. The <alias> will further be used to identify the attribute being exchanged.

Attribute aliases MUST NOT contain newline and colon characters, as specified in the Data Formats / Protocol Messages section of **[OpenID.authentication-2.0]**; they also MUST NOT contain commas (",") and periods (".").

openid.ax.count.<alias>

The number of values returned for the attribute referred to as <alias>.

openid.ax.value.<alias>

Assigns a value to the attribute referred to as <alias>. This parameter format MUST be used if "openid.ax.count.<alias>" is not sent.

openid.ax.value.<alias>.<number>

Assigns a value to the attribute referred to as <alias>. This parameter format MUST be used if "openid.ax.count.<alias>" is sent and at least one value is provided for the associated attribute.

The <number> uniquely identifies the index of the value, ranging from one to the value specified by "openid.ax.count.<alias>". The number of parameters MUST be equal to the value specified by "openid.ax.count.<alias>". The OP is not required to preserve the order of attribute values among fetch responses.

openid.ax.update_url

Returns the "update_url" parameter specified in the request. If the OpenID Provider receives an "update_url" parameter and it intends to support the attribute update feature, it MUST present the "update_url" parameter and value as part of the fetch response message.

A fetch response message may also be sent to the "update_url" specified in **Section 5.1** in response to attribute value updates on the OpenID Provider.

The response to the previous request example, in which the required full name information, and the optional favourite dog information are supplied. Even though three movie names were requested, the OP supplied only two values.

```
openid.ns.ax=http://openid.net/srv/ax/1.0
openid.ax.mode=fetch_response
openid.ax.type.fname=http://example.com/schema/fullname
openid.ax.type.gender=http://example.com/schema/gender
openid.ax.type.fav_dog=http://example.com/schema/favourite_dog
openid.ax.type.fav_movie=http://example.com/schema/favourite_movie
openid.ax.value.fname=John Smith
openid.ax.count.gender=0
openid.ax.value.fav_dog=Spot
openid.ax.count.fav_movie=2
openid.ax.value.fav_movie.1=Movie1
openid.ax.value.fav_movie.2=Movie2
openid.ax.update_url=http://idconsumer.com/update?transaction_id=a6b5c41
```

TOC

6. Store Message

The store message is used to store personal identity information to the OpenID Provider; it provides the means for an RP to transfer to the OP attributes that the user may consider useful, such as by providing them to other RPs. The supported length for attribute aliases MUST be at least 32 characters.

The manner in which the OP processes the attribute payload in a store request if out of scope of this document.

TOC

6.1. Store Request Format

With the exception of "openid.ax.mode", all of the following request fields are OPTIONAL. Any alias referred to in a "openid.ax.value.<alias>" or "openid.ax.value.<alias>.<number>" parameter MUST have an associated "openid.ax.type.<alias>" parameter.

openid.ax.mode

REQUIRED. Value: "store_request".

openid.ax.type.<alias>

The value of this parameter specifies the type identifier URI for an attribute in the sore request. The <alias> will further be used to identify the attribute being exchanged.

Attribute aliases MUST NOT contain newline and colon characters, as specified in the Data Formats / Protocol Messages section of **[OpenID.authentication-2.0]**; they also MUST NOT contain commas (",") and periods (".").

openid.ax.count.<alias>

The number of values sent for the attribute referred to as <alias>. If present, it MUST be greater than zero.

openid.ax.value.<alias>

Assigns a value to the attribute referred to as <alias>. This

Assigns a value to the attribute referred to as <alias>. This parameter format MUST be used if "openid.ax.count.<alias>" is not sent.

openid.ax.value.<alias>.<number>

Assigns a value to the attribute referred to as <alias>. The <number> uniquely identifies the index of the value, ranging from one to the value specified by "openid.ax.count.<alias>". This parameter format MUST be used if "openid.ax.count.<alias>" is sent, and the number of these parameters MUST be equal to the value specified by "openid.ax.count.<alias>".

```
openid.ns.ax=http://openid.net/srv/ax/1.0
openid.ax.mode=store_request
openid.ax.type.fname=http://example.com/schema/fullname
openid.ax.value.fname=Bob Smith
openid.ax.type.fav_movie=http://example.com/schema/favourite_movie
openid.ax.count.fav_movie=2
openid.ax.value.fav_movie.1=Movie1
openid.ax.value.fav_movie.2=Movie2
```

TOC

6.2. Store Response Format

TOC

6.2.1. Storage Success

The successful store operation is indicated by the mode parameter in the store response:

openid.ax.mode

REQUIRED. Value: "store_response_success".

```
openid.ns.ax=http://openid.net/srv/ax/1.0
openid.ax.mode=store_response_success
```

TOC

6.2.2. Storage Failure

A storage failure response has the following format:

openid.ax.mode

REQUIRED. Value: "store_response_failure".

openid.ax.error

OPTIONAL. Parameter describing the error condition leading to the failure response, intended to be presented to the user. The locale of the message should match the locale of the HTTP message.

```
openid.ns.ax=http://openid.net/srv/ax/1.0
openid.ax.mode=store_response_failure
openid.ax.error=General storage failure
```

TOC

7. Security Considerations

OpenID Attribute Exchange is an OpenID extension, and thus uses OpenID Authentication request and response messages for exchanging attributes.

See the "Security Considerations" section of **[OpenID.authentication-2.0]**.

TOC

8. Acknowledgements

John Merrells and other contributors to the document 'draft-merrells-dix'. Portions of that document were re-used for this one.

Mark Wahl advised on how to deal with issues concerning the encoding of attributes.

TOC

9. References

TOC

9.1. Normative References

[OpenID.authentication-2.0]	specs@openid.net, “OpenID Authentication 2.0 - Final,” August 2007 (**TXT**, **HTML**).
[OpenID.value-lang-1.0]	**Wahl, M.**, “**Language Tags for OpenID Values**,” April 2007.
[RFC2119]	**Bradner, S.**, “**Key words for use in RFCs to Indicate Requirement Levels**,” BCP 14, RFC 2119, March 1997 (**TXT**, **HTML**, **XML**).
[RFC3629]	Yergeau, F., “**UTF-8, a transformation format of ISO 10646**,” STD 63, RFC 3629, November 2003 (**TXT**).

TOC

9.2. Non-normative References

[OpenID.axschema]	**Hardt, D.**, “**Schema for OpenID Attribute Exchange**,” May 2007.

TOC

Authors' Addresses

Dick Hardt
Sxip Identity

798 Beatty Street
Vancouver, BC V6B 2M1
CA
Email: dick@sxip.com
URI: http://sxip.com/

Johnny Bufu
Sxip Identity
798 Beatty Street
Vancouver, BC V6B 2M1
CA
Email: johnny@sxip.com
URI: http://sxip.com/

Josh Hoyt
JanRain
5331 SW Macadam Ave. #375
Portland, OR 97239
US
Email: josh@janrain.com
URI: http://janrain.com/